

Remarks**Status of application**

Claims 1-63 were examined and stand rejected in view of prior art, as well as stand rejected for technical issues. The claims have been amended to further clarify Applicant's invention and to address objections raised by the Examiner. Reexamination and reconsideration are respectfully requested.

The invention

Applicant's invention provides for moving information contained in a design time model (e.g., Unified Modeling Language (UML) model) into program source code so as to allow the model representation to remain synchronized with the source code representation. The solution includes means to amend source code to allow inclusion of model information (e.g., UML class diagram). Some of this information is expressed as language constructs. Other information is expressed as additional specifiers to structural code elements ("attributes") or as code artifacts with the express purpose of carrying or grouping model information. The foregoing features operate in conjunction with an engine that - at application run time - can reconstruct the model from information included in the executable application.

When the run-time framework requires the model information from the compiled application, the compiled application is inspected using introspection/reflection mechanisms. Based on run-time inspection of this representation, the run-time framework reconstructs the entire UML-based model at run time. In other words, the solution is able to represent model artifacts in a manner that is reachable by introspection/reflection at run time, thereby allowing the model to be reconstructed at run time. Since the model may be fully represented in the code that is used to compile the application, the model can be restored at run time and used for supporting/implementing the business objects and rules specified (in the model) for the application.

General**A. Oath/Declaration**

In paragraph three of the present action, the Examiner complains that the

Declaration is defective because: "It does not state that the person making the oath or declaration believes the named inventor or inventors to be the joint inventors of the subject matter which is claimed and for which a patent is sought" and "It is not dated by the named inventors." Applicant has reviewed the Declaration, which is PTO standard form PTO/SB/01A (and which was filed with an Application Data Sheet), but has found that the Declaration already includes required appropriate language and that dating by the named inventors is not required by 37 CFR Section 1.63. Via telephone conversation on April 24, 2007, the Examiner indicated that the objection was in error and agreed to withdraw it.

B. Amendment to Specification

Applicant has corrected informalities in paragraphs [0040], [0163] and [0166] of the specification as requested by the Examiner, thereby overcoming the Examiner's objection to informalities in these paragraphs.

In paragraph five of the present action, the Examiner objected to Applicant's specification as containing a computer program listing having over 300 lines on the basis that a computer program listing over 300 lines must be submitted on a source code appendix. Applicant has deleted paragraph [0147] which included a code listing of greater than 300 lines and is electronically filing this same code listing with this amendment in a source code appendix. The source code appendix only contains the code submitted in the text of the original specification and contains no new matter. Applicant has also made amendments to paragraph [0003] of the specification to reflect the additional source code appendix filed with this amendment and has amended paragraphs [0146] and [0148] of the specification based on the deletion of paragraph [0147].

The Examiner also objected to paragraphs [0012] and [0033] of Applicant's specification as containing embedded hyperlinks or browser-executable code. Although Applicant respectfully believes that these paragraphs do not contain any embedded hyperlinks or browser-executable code, Applicant has amended these paragraphs to ensure that no browser-executable code is included so as to overcome the Examiner's objection.

In accordance with the request of the Examiner, Applicant has amended the

specification to include references for several trademarks, including Windows, MacIntosh, Linux, Solaris, Unix, FreeBSD, Pentium, LaserJet, Bluetooth, Solaris and Java when such marks are first referenced in Applicant's specification. However, Applicant takes no position as to the validity or ownership of any such trademarks or whether any such referenced trademarks serve to identify any particular products. In addition, several of the names mentioned by the Examiner are company names (e.g., IBM, Microsoft, Intel and Sun Microsystems) and thus referencing these names as trademarks would not be appropriate.

C. Objections to Claims

The Examiner objected to claims 1-63 as containing an improper claim marker. Applicant notes that the complained-about claim markers (i.e., [c1], [c2], etc.) are caused by the previously referenced PTO stylesheet for electronic filing, over which Applicant has no control. However, these claim markers are eliminated by new listing of claims included in this Amendment, thereby overcoming the objection.

The Examiner also objected to claims 9, 31 and 50 as containing a typographical error. Applicant has amended these claims in the manner suggested by the Examiner, thereby overcoming the objection.

D. Section 112, second paragraph rejection

Claims 4, 17-20, 26, 39, 40-42, 45, 46, and 58-61 stand rejected under 35 U.S.C. 112, second paragraph as being indefinite for failing to distinctly claim the subject matter which Applicant regards as the invention. Applicant has amended claims 4, 17, 18, 20, 26, 39, 40, 42, 45, 58, 59 and 61 to address the issues raised by the Examiner, thereby overcoming the rejection. Applicant respectfully believes that the rejections to claims 19, 41, 46 and 60, which are dependent upon amended claims 18, 40, 45 and 59, are also overcome by these amendments.

E. Section 101 rejection

Claims 22-63 stand rejected under 35 U.S.C. 101 on the basis of non-statutory subject matter.

As to claims 22 and 63, Applicant has amended these claims to reference that the downloadable set of instructions are stored on a computer-readable medium, thereby overcoming the rejection of Applicant's claims 22 and 63 under Section 101.

As to claims 23-42, the Examiner states that Applicant's claimed system constitutes computer programs representing computer listings *per se* and hence are non-statutory. The Examiner references paragraph [0034] at page 15 of Applicant's specification as stating that the corresponding apparatus element may be configured in software or firmware. However, Applicant respectfully believes that the Examiner has incorrectly construed this paragraph (and the balance of Applicant's specification) as stating that the elements of Applicant's invention can only be implemented in software. In fact, Applicant's specification expressly states that the elements may be implemented in hardware, software or firmware (or combinations thereof). This is expressly stated, for example, at paragraph [0034] of Applicant's specification as follows: "...the corresponding apparatus element may be configured in hardware, software, firmware or combinations thereof" (Applicant's specification, paragraph [0034], emphasis added). Applicant's specification also describes in detail a computer hardware and software environment in which Applicant's invention may be implemented (Applicant's specification, paragraphs [0036]-[0046]). Moreover, Applicant states that the software (computer-executable instructions) direct operation of a device under processor control, such as a computer (Applicant's specification, paragraph [0165]). As Applicant's claim defines a useful machine or item of manufacture in terms of a hardware or hardware and software combination, Applicant respectfully believes that it defines a statutory product and overcomes the rejection of claims 23-42 under Section 101.

As to claims 43-63 the Examiner states that these claims are non-statutory as they do not appear to generate a tangible result so as to constitute a practical application as the claims are directed to the act of "making the reconstructed model available." Applicant respectfully believes that the claim limitations of Applicant's claims already generate a tangible result. For instance, Applicant's claim 43 includes limitations of "reconstructing the model from the executable application" and "making the reconstructed model available for supporting operation of the executable application." Although Applicant believes that the original claims generate a tangible result, Applicant has amended

independent claim 43 to add claim limitations of "rendering the reconstructed model for display", thereby overcoming the rejection of claims 43-63 under Section 101.

Prior art rejections

A. Section 102 rejection: Goodwin

Claims 1-6, 12, 15-28, 34, 37, 39-47, 53, and 56-63 stand rejected under 35 U.S.C. 102(b) as being anticipated by US Patent 6,199,195 to Goodwin et al (hereinafter "Goodwin"). The Examiner's rejection of Applicant's claim 1 as follows is representative of the Examiner's rejection of these claims as anticipated by Goodwin:

As per Claim 1, Goodwin et al. disclose:

- creating a model describing business objects and rules of the application (see *Column 6: 37-44, "The system and method will also allow developers to generate objects based on a framework of services they author by composing services based on the object templates into objects that support the composed behaviors and methods. This is accomplished through the code generator 210, which interfaces with the unified models 206, which are expressed in a unified modeling language, such as Unified Modeling Language (UML). "*);
- creating source code for the application, including representing the model within the source code itself (see *Column 13: 52-55, "Thus, the code generator 330 can support the creation of, for example, IDL, JAVA or C++ files ... "*);
- compiling the source code into an executable application (see *Column 14: 34-40, "Outputs from the code generator 404 include Interface Definition Language (IDL) files 422 (*.idl), which are processed by an idl-to-Java compiler, e.g., Visibroker's IDL2JA VA, for the generation of CORBA ORB services ... "*);
- running the executable application on a target computer in conjunction with a run-time framework that provides services to the executable application (see *Column 15: 45-65, "The data server 332 operates at run time ... "and "When deployed within a client application, the data server 332 launches, starts, manages and controls execution of a set of services. "*); and
- while the executable application is running, reconstructing the model from the executable application and making it available to the run-time framework (see *Column 16: 36- 64, "The unified models contain mappings about the data sources of each of the classes. " and "Each of the query managers that allows modification of information (insert, update or delete) must support distributed transactions." and "The data server 332 employs the unified models to provide the Clients 338 periodically updated query-based views of distributed heterogeneous*

databases.").

Under Section 102, a claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in the single prior art reference. (See, e.g., MPEP Section 2131.) As will be shown below, Goodwin fails to teach each and every element set forth in Applicant's claims 1-6, 12, 15-28, 34, 37, 39-47, 53, and 56-63 (as well as other claims), and therefore fails to establish anticipation of the claimed invention under Section 102.

At the outset, Applicant does not claim to have invented the notion of using a model for generating application source code; Applicant acknowledges that using a model for generating source code is taught by Goodwin as well as other prior art references. Applicant's invention, however, does not simply generate application code based on a model, but rather provides for representing the entire model inside the code in a manner that enables the model to be reconstructed from the executable application code at runtime. As discussed below in more detail, Goodwin uses a model to generate code, but does not teach incorporating the model into the source code of the application or reconstructing the model from the compiled code of the application.

Applicant's invention provides for representing the original model (e.g., UML model) inside source code generated from the model. Applicant's invention ensures that everything in the model is represented in the emitted source code, so that it can later be reconstructed from the code to a model again (Applicant's specification, paragraphs [0060] - [0061]). This includes not only representing portions of the UML model having a natural representation in the source code (e.g., class names), but also involves adding attributes and other information to the code for conveying meta information about the UML model which is not otherwise expressible in the code (Applicant's specification, paragraph [0070]; Fig. 3D). As an example, consider an association relationship such as a residency association linking Person and Building. When it comes to associations such as this residency association, a standard code representation of such association only captures a fragment of what is represented in a UML model with respect to the association element. With Applicant's invention, however, attributes are used to express important information about the model (e.g., an association element of the model) that cannot be expressed in normal structural coding elements (Applicant's specification,

paragraphs [0071] - [0081]). Applicant's solution provides for creating a representation of the model in the source code so that all information from the model can be expressed in the code and subsequently made available at runtime (Applicant's specification, paragraph [0081]). Thus, Applicant's claimed invention incorporates a full representation of the model into the source code itself. This enables the entire model that was used for generating the code to be fully reconstructed back into model form at runtime from the executable code itself as hereinafter described in more detail. These distinctive features are specifically described in Applicant's claims. For example, Applicant's amended claim 1 includes the following claim limitations:

In a computer system, an improved method for developing and executing an application, the method comprising:
creating a model describing business objects and rules of the application;
creating source code for the application, including representing the model within the source code itself;

(Applicant's amended claim 1, emphasis added)

Although Goodwin describes generating code from a model, Applicant's review of Goodwin finds no teaching of representing the model within the source code itself as provided in Applicant's specification and claims. Goodwin's system provides instead for storing unified models in a schema repository accessed by a schema server (Goodwin, col. 8, lines 58-62; Fig 3). The schema repository of Goodwin's system is a SQL-compliant database (Goodwin, col. 10, lines 38-41). Thus, Goodwin's approach relies on persisting a design-time representation of the model in a database instead of incorporating the model into the application code itself.

Another significant difference between Applicant's invention and the system of Goodwin is that Applicant's invention provides for reconstructing the model at runtime from the executable code of the application (Applicant's specification, paragraph [0061]). This feature of reconstructing the model from the executable code at runtime is only possible because of the prior step of incorporating a representation of the model in the source code. As Goodwin's solution does not represent the model in the code, it also cannot reconstruct the model from the code as doing so requires that the necessary representation of the model is included in the code in the first place.

With Applicant's solution, one need not have the original UML model available (e.g., persistently stored in a repository) at the point the model is reconstructed at runtime. Instead, Applicant's invention enables the original model to be reconstructed at runtime from the compiled binary using introspection/reflection mechanisms (Applicant's specification, paragraphs [0060]-[0061]). This feature of reconstructing the model from the executable application is specifically indicated in Applicant's claims, including, for example, the following limitations of Applicant's claim 1:

compiling the source code into an executable application;
running the executable application on a target computer in conjunction with a run-time framework that provides services to the executable application; and
while the executable application is running, reconstructing the model from the executable application and making it available to the run-time framework.

(Applicant's claim 1, as amended, emphasis added)

Goodwin has no comparable teaching of reconstructing the model from the executable application at run time. Instead, Goodwin's system relies on retrieving model information through the schema server by a dataserver (Goodwin, col. 10, lines 63-65). The dataserver of Goodwin's system does not reconstruct the model from executable code. Instead, it simply consumes model information from the schema server. The schema server itself doesn't reconstruct any model. The schema server is fed at design time with the original unified model that was used to generate code and the model is persisted by the schema server. In other words, the dataserver of Goodwin's system simply retrieves a stored copy of the model generated at design time from a repository. This is not equivalent to reconstructing the model from the executable application (compiled code) as provided in Applicant's specification and claims.

Advantages of Applicant's claimed invention of storing the model in the compiled code include that the model information and the code are synchronized. With Applicant's approach there is only one representation of the model. In the system of Goodwin, in contrast, there may be a mismatch between the compiled code and the stored model which was generated and persisted in the schema repository. The code and the stored model may, in fact, represent different versions from different points in time. For example, the code may have been modified after the model was created and stored at

design time. Alternatively, the model stored in the repository of Goodwin's system may have been updated, but the generated code may not reflect the most recent updates to the model. With Applicant's approach of storing the model in the compiled code, there is only one representation of the model which is synchronized with the code as the model is represented inside the code itself.

All told, Goodwin provides no teaching of fully representing the model inside the source code so as to enable reconstructing the model from the executable code at runtime as described in Applicant's specification and claims. Therefore, as Goodwin does not teach or suggest all of the claim limitations of Applicant's claims 1-6, 12, 15-28, 34, 37, 39-47, 53, and 56-63 (and other claims) it is respectfully submitted that the claims distinguish over this reference and overcome any rejection under Section 102.

B. Section 103: Goodwin in view of Mullins

Claims 7, 8, 29, 30, 48 and 49 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin (above) in view of US Patent 7,103,600 to Mullins (hereinafter "Mullins"). Here, the Examiner relies on Goodwin as substantially teaching the claimed invention (as per the Examiner's rejection under Section 102 above), but acknowledges that Goodwin does not disclose use of reflection for reconstruction of the model from the executable application and adds Mullins for this teaching.

As to these claims, Applicant believes that the claims are allowable for at least the reasons set forth above in the response to the Section 102 rejection as to the deficiencies of the Goodwin reference with respect to Applicant's invention. In particular, Goodwin does not teach fully representing the model within the application code as discussed above. Although Mullins does discuss use of reflection in a system for persisting all or part of a complex data object graph model, it does not cure any of the above-described deficiencies of Goodwin as it provides no comparable teaching of representing the model within application code generated from the model or of reconstructing the model from the executable application at runtime. The teachings of Mullins referenced by the Examiner describe use of runtime classes to persist instances of a compiled application for use in developing a navigation model for the application (Mullins, col. 35, lines 38-43; col. 36 lines 50-54). However, even assuming this is somehow analogous to Applicant's

invention for reconstructing a unified model from executable code, Mullins' system cannot reconstruct a full model from the executable application as the necessary steps to represent the model inside the code have not been undertaken in the first place. In addition, Mullins provides for persisting complex data object models to a data store or information repository external to the application code (Mullins, col. 11, lines 4-27), while with Applicant's approach the model is stored inside the executable code itself.

As Goodwin and Mullin references, even when combined, do not teach or suggest all the claimed features of Applicant's invention, it is respectfully submitted that Applicant's claimed invention as set forth by these claims is distinguishable over the two references, and that the rejection under Section 103 is overcome.

C. Section 103: Goodwin, Mullins and Schloegel

Claims 9, 31 and 50 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin (above) in view of Mullins (above), further in view of US Pub. Application 2004/0044990 of Schloegel et al (hereinafter "Schloegel"). Here, the Examiner relies on Goodwin and Mullins as substantially teaching the claimed invention (as per the Examiner's rejection under Section 103 above), but acknowledges that Goodwin and Mullins do not disclose traversal using a selected one of depth-first, breadth-first and ad-hoc traversal techniques and adds Schloegel for these teachings.

Applicant believes that these claims are allowable for at least the reasons set forth above in the response to the Section 102 and 103 rejections as to the deficiencies of the Goodwin and Mullins reference with respect to Applicant's invention. Although Schloegel describes alternative techniques for traversal during code generation, it does not include any teaching of representing a unified model within application code generated from the model or of reconstructing the model from the executable application at runtime. Thus, it does not cure any of the above-described deficiencies of Goodwin and Mullins as to Applicant's invention. As the combined references do not teach or suggest all the claimed features of Applicant's invention, it is respectfully submitted that Applicant's claimed invention as set forth by these claims is distinguishable over the combined references, and that the rejection under Section 103 is overcome.

D. Section 103: Goodwin in view of Baisley

Claims 10, 11, 32, 33, 51 and 52 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin (above) in view of US Patent 6,711,734 to Baisley (hereinafter "Baisley"). The Examiner again relies on Goodwin as substantially teaching the claimed invention (as per the Examiner's rejection under Section 102 above), but acknowledges that Goodwin does not disclose creating a UML package for the reconstructed model based on detecting a class having a package element or an attribute specifying that a class belongs to a UML package. The Examiner adds Baisley for these teachings.

Applicant believes that the claims are allowable for at least the reasons set forth above in the response to the Section 102 rejection as to the deficiencies of the Goodwin reference with respect to Applicant's invention. Baisley describes automatically translating a stored Metaobject Facility (MOF) metamodel into a Unified Modeling Language (UML) Model (Baisley, abstract). However, Baisley does not provide any teaching of reconstructing a UML model from an executable application nor does it provide any teaching of incorporating the model into the source code of the application so as to enable the subsequent reconstruction of the model as described in Applicant's specification and claims. Accordingly, as Baisley does not cure any of the above-described deficiencies of Goodwin, Applicant respectfully submits that Applicant's claimed invention is distinguishable over the combined references, and that the rejection under Section 103 is overcome.

E. Section 103: Goodwin in view of Mutschler

Claims 13, 14, 35, 36, 54 and 55 stand rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin (above) in view of US Patent 7,162,462 to Mutschler, III (hereinafter "Mutschler"). The Examiner again relies on Goodwin as substantially teaching the claimed invention (as per the Examiner's rejection under Section 102 above), but acknowledges that Goodwin does not disclose constructing a common superclass for classes of the reconstructed model and adds Mutschler for this teaching.

Applicant believes that the claims are allowable for at least the reasons set forth

above in the response to the Section 102 rejection as to the deficiencies of the Goodwin reference with respect to Applicant's invention. Mutschler does not cure any of these deficiencies as it simply describes the creation of a superclass for both events and rules, allowing them to share data and methods relating to their common dynamic behavior over time (Mutschler, col. 5, lines 40-43) in the context of a system providing time sensitivity to an inference engine (Mutschler, abstract). Accordingly, as the combined references do not teach or suggest all the claimed features of Applicant's invention, it is respectfully submitted that Applicant's claimed invention is distinguishable over the combined references, and that the rejection under Section 103 is overcome.

Any dependent claims not explicitly discussed are believed to be allowable by virtue of dependency from Applicant's independent claims, as discussed in detail above.

Conclusion

In view of the foregoing remarks and the amendment to the claims, it is believed that all claims are now in condition for allowance. Hence, it is respectfully requested that the application be passed to issue at an early date.

If for any reason the Examiner feels that a telephone conference would in any way expedite prosecution of the subject application, the Examiner is invited to telephone the undersigned at 925 465-0361.

Respectfully submitted,

Date: May 23, 2007

/G. Mack Riddle/

G. Mack Riddle, Reg. No. 55,572
Attorney of Record

925 465-0361
925 465-8143 FAX